Spherical Superpixel Segmentation

Qiang Zhao, Feng Dai, *Member, IEEE*, Yike Ma, *Member, IEEE*, Liang Wan, *Member, IEEE*, Jiawan Zhang, *Member, IEEE*, and Yongdong Zhang, *Senior Member, IEEE*

***Abstract*—These days, superpixel algorithms are widely used in computer vision and multimedia applications. However, existing algorithms are designed for planar images, which are less suited to deal with wide angle images. In this paper, we present a superpixel segmentation method for 360° spherical images. Unlike previous methods, our approach explicitly considers the geometry for spherical images and makes clustering to spherical image pixels. It starts with the seeds defined by Hammersley points sampled on the sphere, then iterates between assignment step and update step, which are both based on the distance metric respecting spherical geometry. We evaluate our method on the transformed Berkeley segmentation dataset and panorama segmentation dataset collected by ourselves. Experimental results show that our method can gain better performance in terms of adherence to image boundaries and superpixel structural regularity. Furthermore, superpixels generated by our method can reserve the coherence across image boundaries and all have closed contours.**

***Index Terms*—Superpixel, segmentation, spherical image, panorama, clustering, Hammersley, SLIC.**

I. Introduction

Superpixel generation has become a key building block of many computer vision and multimedia applications, such as gesture recognition [1], [2], image parsing [3], [4], depth estimation [5], [6], segmentation [7]–[9], object localization [10], [11] and saliency detection [12], [13]. The core idea of superpixel generation is grouping similar pixels into perceptually meaningful atomic regions, which always conform well to the local image structures. The pixels in each of these regions are likely to belong to the same real world object. Hence superpixels provide a better spatial support for feature extraction than conventional rectangular windows. Another advantage of superpixel generation is that it can significantly decrease the computational complexity of subsequent tasks since the number of superpixels is dramatically less than that of pixels. Respecting the aforementioned desired properties of superpixels, a lot of superpixel algorithms have been proposed in the literature. Early methods that use pixel color statistics, e.g. mean shift [14] or the graph based method [15], often produce superpixels of highly irregular shape and size, which can lead to undersegmentation. Later many researchers realize the importance of compactness constraints [16]–[19], and propose methods to cope with undersegmentation. Compactness is also used to better preserve the spatial and topological structure of the original images [17], [19], [20].

As a key component, superpixel generation is also widely used in applications involving spherical (panoramic) images [21]–[24]. Yet, in these applications existing superpixel algorithms designed for planar images are simply used to produce spherical superpixels, which may have three problems. First, the image sphere is used to model the visual information seen from a single viewpoint. For representation, we must flat the image sphere and map it onto an image plane. However any such mapping inevitably introduces perceptual distortions [25], which will degrade the performance of superpixel algorithms. Second, full spherical images capture 360° field-of-view of the surrounding environment, which forms a closed surface. Therefore the superpixels of a spherical image should have closed contours when mapping to the sphere, but this is not guaranteed by the planar methods. Third, regular shape and size are also preferred in spherical superpixel segmentation. Although some planar methods are designed to generate compact and regular superpixels, e.g. SLIC [19], the uniformity would be lost if we apply them on spherical images. All of these problems are caused by the lack of considering the *geometry* for spherical images.

In this paper, we propose a spherical superpixel generation algorithm that effectively handles the above problems. Our method resembles the fundamental idea of the mature SLIC superpixel algorithm [19] and uses clustering to generate superpixels by explicitly considering the geometry for spherical images. Specifically we first use Hammersley points [26] sampled on sphere to initialize superpixel centers. Then, cosine dissimilarity and spherical distance are used as the spatial distance measure when assigning pixels and updating the superpixel centers. For quantitative evaluation, we transform the Berkeley segmentation dataset to the spherical domain and collect a new panorama segmentation dataset. Our method can achieve better boundary

Manuscript received January 24, 2017; revised July 5, 2017; accepted October 20, 2017. Date of publication November 13, 2017; date of current version May 15, 2018.This work was supported in part by the National Natural Science Foundation of China under Grants 61327013, 61572354, 61525206, 61702479, and 61771458, and in part by Key Research Program of the Chinese Academy of Sciences under Grant KFZD-SW-407. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ivan V. Bajic. *(Corresponding author: Liang Wan.)*

Q. Zhao is with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with Tianjin University, Tianjin 300072, China (e-mail: zhaoqiang@ict.ac.cn).

F. Dai, Y. Ma, and Y. Zhang are with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: fdai@ict.ac.cn; ykma@ict.ac.cn; zhyd@ict.ac.cn).

L. Wan and J. Zhang are with the School of Computer Software, Tianjin University, Tianjin 300072, China (e-mai: lwan@tju.edu.cn; jwzhang@tju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2772842

adherence and structural regularity on these datasets. Experiments on real captured spherical images are also carried out for visual comparison and the superior performance of our method is validated.

The remaining of this paper is organized as follows. Section II reviews the most related work including both planar and spherical superpixel generating methods. We briefly describe the geometry for spherical images in Section III, which is followed by the details of our algorithm in Section IV. Section V comprehensively evaluates the proposed method and state-of-the-art competitors. We also discuss the influence of parameters for our method. Section VI concludes the paper and gives the possible directions of future work.

II. Related Work

In this section, we will give the related work from two aspects: superpixel algorithms designed for planar images and the methods used to generate superpixels for spherical images.

A. Planar Superpixel Algorithms

Superpixel algorithms aim to over-segment the image by grouping similar pixels that belong to the same object. Comaniciu and Meer [14] apply mean shift, an iterative procedure for locating local maxima of a density function, to find modes in the color feature space of an image. Pixels that converge to the same mode thus define the superpixels. However this method is relatively slow, and can not control the amount or size of superpixels. Felzenszwalb and Huttenlocher [15] present a graph based image segmentation method, in which pixels are vertices of the graph and dissimilarities between neighboring pixels are edges. Agglomerative clustering of pixels is performed such that each superpixel is the minimum spanning tree of the graph. Their method adheres well to image boundaries, but produces superpixels with very irregular sizes and shapes. The TurboPixels algorithm [16] segments an image into superpixels by dilating a set of seed locations using geometric flow. This method can produce a lattice-like structure, and the produced superpixels are constrained to have uniform size, compactness, and boundary adherence. Veksler *et al.* [17] propose a superpixel algorithm using global optimization. In their approach, superpixels are obtained by stitching together overlapping image patches such that each pixel belongs to only one of the patches. Zeng *et al.* [18] describe a structure-sensitive over-segmentation technique, which is formulated as an energy minimization with the geodesic distance. It considers the homogeneity of image appearance and compactness constraints; thus it can greatly limit under-segmentation. Achanta *et al.* [19] introduce simple linear iterative clustering (SLIC), which adapts a k-means clustering approach to efficiently generate superpixels with regular size and shapes. Starting from an initialized superpixel partition, SEEDS [27] exchanges pixels between superpixels iteratively. This algorithm defines an energy function by enforcing homogeneity of the color distribution within superpixels, which can be efficiently evaluated by hill-climbing optimization.

B. Superpixels for Spherical Images

To generate superpixels for spherical images, existing planar superpixel algorithms are usually used directly. Cabral and Furukawa [23] present a system to reconstruct floorplan from images. Indoor spherical images are segmented into superpixel [15], whose texture homogeneity is exploited for structure classification. Sakurada and Okatani [24] propose a change detection method, where superpixel segmentation of spherical images is integrated with a low resolution change map estimated from CNN features to get precise boundaries of the changes. To mitigate the effects of image distortions, spherical images can be transformed into piecewise perspective images first, and then superpixels are generated from these perspective images. With this strategy, a superpixel based multi-view stereo method has been developed in [21] to deal with panoramic sequences, which assigns one depth per superpixel. Košecka [22] adopts a similar idea to over-segment the panorama into superpixels. Each superpixel is labeled as changed or unchanged after reconstructing a coarse 3D model and aligning the image to the model. Despite the success of above works, they suffer from some or all of the problems described in the first section.

A panoramic over-segmentation algorithm is proposed in [28] by extending the graph based method [15]. Three modifications are made to the original algorithm to make it suitable for spherical images: additional adjacency relationships between image pixels are considered; different sizes are assigned to pixels at different heights; a post-process is applied to remove thin superpixels. The algorithm can reserve the coherence across the boundary of spherical image, and create superpixels with closed contours. In the preliminary conference version of this work [29], we present a spherical superpixel segmentation method, which is based on the idea of SLIC. Contrasted with Yang's method [28], it incorporates the compactness constraints and can decrease undersegmentation. The superpixels generated by this method also have regular shape and size.

III. Geometry for Spherical Images

Our method is designed to deal with spherical images. Before introducing the details of our algorithm, we first describe the geometry for these images.

Equirectangular projection is used to represent the spherical images. This projection maps meridians and circles of latitude of image sphere to vertical and horizontal coordinates of spherical image respectively. Thus the width w of spherical image is twice the height h. For a given point $[X\ Y\ Z]$ on the unit sphere, the 2D image coordinates $[x\ y]$ of its corresponding pixel can be determined by two steps. First the spherical coordinates of the 3D point is obtained by

$$\begin{cases} \theta = \arctan 2(Y, X) \\ \phi = \arccos(Z) \end{cases} \tag{1}$$

where θ is the azimuthal angle and ϕ is the polar angle as shown in Fig. 1. The function $\arctan 2$ in the equation is the quadrant-aware version of $\arctan$. The 2D image coordinates $[x\ y]$ on the

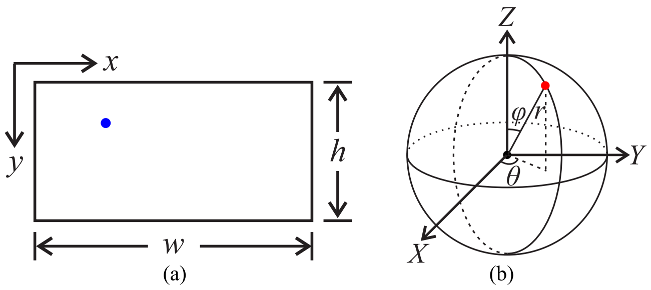


Fig. 1. The geometry for spherical images: (a) we represent the spherical image with Equirectangular projected image, whose resolution is $w \times h$ pixels; (b) given a 3D point on the sphere (red point), we can find its corresponding image coordinates (blue point in (a)) on the Equirectangular projected spherical image.

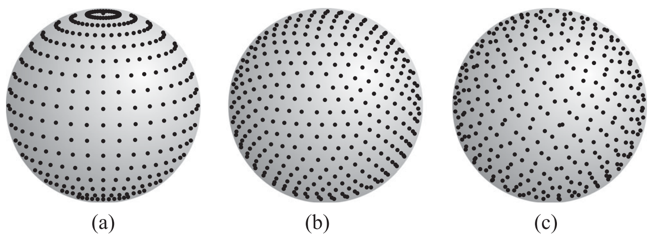


Fig. 2. Sampling points on the sphere with different methods: (a) latitude-longitude sampling (b) geodesic sampling and (c) Hammersley sampling.

spherical image plane is given by

$$\begin{cases} x = \frac{\theta w}{2\pi} \\ y = \frac{\phi h}{\pi} \end{cases} \tag{2}$$

after the range of θ is mapped from $(-\pi, \pi]$ to $[0, 2\pi)$.

The reverse transformation of the above procedure can be used to find the corresponding 3D positions of spherical image pixels.

IV. Spherical Superpixels

Our method is an algorithm in the spirit of SLIC [19], which adopts the k-means clustering algorithm for superpixel generation. It first initializes the cluster centers (or superpixel centers) by sampling the unit sphere. Then it iterates between the assignment step, which associates each pixel to its nearest cluster center, and the update step, which adjusts the cluster centers. All these operations are carried out in the spherical domain to respect the geometry of spherical images.

A. Initialization

The purpose of the initialization step is giving the seed points for cluster centers, which is required by the following assignment step. To guarantee that the seed points are uniformly distributed on the image domain, the planar superpixel algorithm [19] samples these points on a regular spaced grid. Because the polar regions of the spherical images are over sampled as shown in Fig. 2(a), the seed points can not be sampled directly on the equirectangular image plane. Instead in our approach the cluster centers $C_i = [X_i\ Y_i\ Z_i\ l_i\ a_i\ b_i]$, $i = 1, \ldots, k$ are initialized by uniformly sampling the unit sphere, where $[X_i\ Y_i\ Z_i]$ is the 3D unit vector describing the sampling position and $[l_i\ a_i\ b_i]$ is the LAB color of the spherical image pixel at the sampling point. The LAB color space is used in our work, as it is perceptually uniform [30] and is widely used in the state-of-the-art superpixel algorithms [18], [19], [27]. Given the sampling position $[X_i\ Y_i\ Z_i]$, the image coordinate of the pixel can be computed using (1) and (2). To avoid putting a superpixel on an edge, we move the initial centers to the locations that have lowest gradient in a 3×3 neighborhood. By using the gradient in spherical coordinate system, the gradient of the spherical image I is computed as

$$\nabla I = \left[\frac{1}{\sin\phi}\frac{\partial I}{\partial \theta}, \frac{\partial I}{\partial \phi}\right]. \tag{3}$$

To sample the sphere uniformly, there are two commonly used methods. The first one is geodesic sampling in Fig. 2(b), which is based on recursive subdivision of icosahedron. The second method is Hammersley sampling in Fig. 2(c). The Hammersley sampling [31] first generates Hammersley points in 2D unit square based on the Van der Corput sequence [32]. Then these points are mapped to the unit sphere via linear scaling and z-preserving radial projection. From Fig. 2 we can see that geodesic sampling can give more uniformly distributed sampling points. In our work, we adopt Hammersley sampling for two reasons. First, Hammersley sampling can generate arbitrary number of sampling points, while geodesic sampling is less flexible and can only generate $5 \times 2^{2n+1} + 2$ sampling points, where n is the subdivision level. Although more sophisticated geodesic sampling exists [33], the available sampling resolutions are still restricted. The second reason is that we do not need such uniformly distributed sampling points of geodesic sampling. This is because these sampling points (superpixel centers) will be updated in the following iteration steps. We also find that geodesic sampling and Hammersley sampling would give approximately the same performance through experiment.

B. Iteration

Given initialized cluster centers, the assignment step associates each pixel $p = [X\ Y\ Z\ l\ a\ b]$[1] to its nearest cluster center. Compared to conventional k-means clustering, where pixel p is compared with all cluster centers C_i, our approach only considers cluster centers that are located in the neighborhood of pixel p [19]. This is equivalent to limit the search space of each cluster center to a region R_i of size $2S \times 2S$, where S is the superpixel size. For the spherical images, we assume all superpixels have size

$$S = \sqrt{\frac{A}{k}} = \sqrt{\frac{4\pi r^2}{k}} = \sqrt{\frac{4\pi(\frac{w}{2\pi})^2}{k}} = \frac{w}{\sqrt{k\pi}} \tag{4}$$

[1]Note that we use p to denote a pixel or its feature vector. The meaning which one should be chosen is determined from the context.

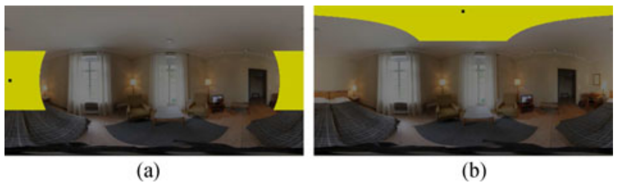


Fig. 3. Local search region when image coordinates fall outside of valid range: the yellow regions represent the local search region and the black points are superpixel centers. Note that the resolution of spherical image is 256 by 512 pixels and search region size is set as 50 pixels for illustration purpose.

where A and r are the area and radius of the sphere respectively. For cluster i, the local search region R_i is defined as

$$R_i = \{[x\ y] | x_i - \frac{S}{\sin\phi} \le x \le x_i + \frac{S}{\sin\phi}, \\ y_i - S \le y \le y_i + S\}, \tag{5}$$

where $[x_i\ y_i]$ is the 2D image coordinates for the center of the i-th cluster computed from $[X_i\ Y_i\ Z_i]$, $\phi = \frac{y\pi}{h}$ is the polar angle corresponding to the y-th row of the image by the inverse transform of (2). Since the north and south poles of image sphere are mapped to the top and bottom boundaries of the spherical image respectively by Equirectangular projection, the mirror texture address mode should be used if y coordinate of the search region is outside of the valid range. To be specific, the search region for y becomes

$$\begin{cases} 0 \le y \le y_i + S, & \text{if } y_i - S < 0 \\ y_i - S \le y \le h - 1, & \text{if } y_i + S \ge h \\ y_i - S \le y \le y_i + S, & \text{otherwise.} \end{cases} \tag{6}$$

Because the left and right boundaries of the spherical image correspond to the same meridian, the wrap texture address mode should be used if x coordinate of the search region falls outside of the valid range. For example if the search region is close to the left boundary of the spherical image, i.e. $x_i - \frac{S}{\sin\phi} < 0$, the range of x becomes

$$x \in \left[0, x_i + \frac{S}{\sin\phi}\right] \cup \left[x_i - \frac{S}{\sin\phi} + w, w - 1\right], \tag{7}$$

and if the search region is close to the right boundary of the spherical image, i.e. $x_i + \frac{S}{\sin\phi} \ge w$, the range of x becomes

$$x \in \left[0, x_i + \frac{S}{\sin\phi} - w\right] \cup \left[x_i - \frac{S}{\sin\phi}, w - 1\right]. \tag{8}$$

Fig. 3 shows exemplar search regions when x and y coordinates are outside of valid ranges. Because the polar regions of the spherical images have more distortions than the central region, the local search region in Fig. 3(b) is stretched more seriously than that in Fig. 3(a).

Given the definition of the local search region, the assignment step can be achieved by the following minimization problem

$$L(p) = \underset{i|[x\ y]\in R_i}{\arg\min}\, D(C_i, p), \tag{9}$$

where $D(C_i, p)$ is the distance measure that will be discussed in Section IV-C.

Once each pixel has been associated to the nearest cluster center, an update step is taken to adjust each cluster center to be the point that minimizes the sum of distances between itself and all the pixels belonging to the cluster under distance measure D, i.e.

$$C_i = \underset{C_i}{\arg\min} \sum_{L(p)=i} D(C_i, p). \tag{10}$$

The assignment and update steps are repeated in turn until the algorithm converges or some specified number of iterations is exceeded.

C. Distance Measure

The distance measure $D(C_i, p)$ computes the distance between cluster center C_i and pixel p. It combines color distance d_c and spatial distance d_s, and is defined as

$$D(C_i, p) = \frac{d_c(C_i, p)}{N_c} w_c + \frac{d_s(C_i, p)}{N_s}, \tag{11}$$

where the denominators N_c and N_s are the maximum color and spatial distances of pixels to their cluster centers after each iteration, w_c is the parameter that allows us to weigh the relative importance between color distance and spatial distance. The definition of color distance is the same as that of SLIC algorithm [19]. It is given by the LAB color distance

$$d_c(C_i, p) = (l - l_i)^2 + (a - a_i)^2 + (b - b_i)^2 \tag{12}$$

between superpixel C_i and pixel p. The spatial distance is different. Because the cluster centers and all the pixels are located on the sphere, the Euclidian distance between $[X\ Y\ Z]$ and $[X_i\ Y_i\ Z_i]$ is not suited. In this paper, we use cosine dissimilarity for spatial distance d_s and term this method as *Cos-SphSLIC*. Denote the angle between $[X\ Y\ Z]$ and $[X_i\ Y_i\ Z_i]$ as α. The cosine dissimilarity or d_s can be computed as

$$\begin{aligned} d_s(C_i, p) &= 1 - \cos\alpha \\ &= 1 - \frac{\langle [X\ Y\ Z], [X_i\ Y_i\ Z_i] \rangle}{\|[X\ Y\ Z]\|_2 \|[X_i\ Y_i\ Z_i]\|_2} \\ &= 1 - \langle [X\ Y\ Z], [X_i\ Y_i\ Z_i] \rangle, \end{aligned} \tag{13}$$

where $\langle \mathbf{a}, \mathbf{b} \rangle$ is the dot product of vector $\mathbf{a}$ and $\mathbf{b}$, and the last equality holds due to the fact that $[X\ Y\ Z]$ and $[X_i\ Y_i\ Z_i]$ are both located on the unit sphere. This forms a hybrid algorithm, where color component is the cartesian k-means clustering and spatial component is the spherical k-means clustering [34]. Compared with cosine dissimilarity, the spherical distance (or known as great circle distance) is a more natural way to measure the distance between two points on the sphere. To show whether this widely used distance will give better performance, we adopt another spatial distance definition

$$d_s(C_i, p) = \arccos(\langle [X\ Y\ Z], [X_i\ Y_i\ Z_i] \rangle). \tag{14}$$

Our method with this distance is termed as *Avg-SphSLIC*, and we will compare its performance with that of Cos-SphSLIC in Section V.

The assignment step and update step both involve minimization problems. The problem in assignment step can be solved simply by exhaustive test. Note that the above two spatial distance will give us the same assignment result as (13) and (14) have the same montonicity with respect to C_i. The update step is a little tricky. Because the color distance and spatial distance are independent, we can optimize them individually. Minimization with respect to color distance d_c gives us the color component of the new cluster center as the mean over $[l\ a\ b]$ vectors of all the pixels belonging to the cluster. When cosine dissimilarity is used, the spatial part of (10) becomes

$$\begin{aligned}
&\underset{[X_i\ Y_i\ Z_i]}{\arg\min} \sum_{L(p)=i} (1 - \langle [X\ Y\ Z], [X_i\ Y_i\ Z_i] \rangle) \\
&= \underset{[X_i\ Y_i\ Z_i]}{\arg\min} \Big(\sum_{L(p)=i} 1 - \langle \sum_{L(p)=i} [X\ Y\ Z], [X_i\ Y_i\ Z_i] \rangle \Big) \\
&= \underset{[X_i\ Y_i\ Z_i]}{\arg\max} \langle \sum_{L(p)=i} [X\ Y\ Z], [X_i\ Y_i\ Z_i] \rangle \\
&= \frac{\sum_{L(p)=i} [X\ Y\ Z]}{\| \sum_{L(p)=i} [X\ Y\ Z] \|_2}
\end{aligned} \tag{15}$$

where the last equality follows from the property of dot product that $\langle \mathbf{a}, \mathbf{b} \rangle = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2$ if and only if the angle between vector $\mathbf{a}$ and $\mathbf{b}$ is zero and the fact that $[X_i\ Y_i\ Z_i]$ is a unit vector. Different from Cos-SphSLIC, the update step of Avg-SphSLIC needs to find a point on the sphere such that the sum of spherical distances between this point and the pixel positions is minimum. Unfortunately, there is not analytic form for this problem. In this paper, we adopt a spherical average algorithm [35], which starts with an initial estimate and produces better estimates iteratively.

V. Experimental Results

In this section, we first make quantitative evaluation of our methods, SLIC [19], efficient graph based segmentation (EGS) [15] and panorama version of EGS (PanoEGS) [28]. Then we qualitatively compare different methods. Finally, we give their timing performance followed by a discussion about Cos-SphSLIC and Avg-SphSLIC. All these methods are applied on the equirectangular projected spherical images, which have image distortions.

A. Quantitative Evaluation

1) Metrics: One of the most important properties of superpixels is adherence to image boundaries [19]. We use three standard metrics to evaluate boundary adherence of different algorithms: boundary recall [18], [36], [37], under segmentation error [16], [17], [19] and achievable segmentation accuracy [27], [37], [38]. In the following, we first describe these metrics for the sake of completeness, then give their formulation for the spherical case. For convenience, we use $\mathcal{G} = \{g_1, g_2, \ldots, g_m\}$ to represent the ground truth segmentation, and use $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ to represent a superpixel segmentation.

Boundary recall (BR) measures what fraction of the ground truth edges coincide with the boundary of superpixels. We compute it as

$$BR(\mathcal{G}, \mathcal{S}) = \frac{\sum_{p \in B(\mathcal{G})} A(p) I(\min_{q \in B(\mathcal{S})} d_s(p, q) < \varepsilon)}{A(B(\mathcal{G}))}, \tag{16}$$

where $B(\mathcal{G})$ and $B(\mathcal{S})$ are the union sets of ground truth segmentation boundaries and computed superpixel boundaries respectively, d_s is the cosine dissimilarity in (13). Function $I(\cdot)$ is an indicator function that checks whether there are superpixel boundaries falling within the neighborhood of ground truth edges. $A(\cdot)$ is the surface area of a pixel set, whose calculation is given in Appendix A. Compared with standard boundary recall, two differences are introduced in the above formula to make this metric applicable to spherical images: first, we use cosine dissimilarity in the neighborhood definition to account for spherical geometry; second, the cardinality of the boundaries set is replaced with surface area, so that the boundaries near the polar regions will not be over-weighted.

Under segmentation error (USE) measures how many pixels from superpixels overlapping a ground truth segment leak across the boundaries. It is based on the fact that a superpixel should not overlap more than one object. Given the ground truth segmentation and generated superpixels, we quantify the under-segmentation error as

$$USE(\mathcal{G}, \mathcal{S}) = \frac{\sum_i (\sum_{k | s_k \cap g_i \neq \emptyset} A(s_k) - A(g_i))}{\sum_i A(g_i)}. \tag{17}$$

Note that the summed area of all ground truth segments $\sum_i A(g_i)$ may not be equal to the surface area of sphere. This is because the input image may not cover the full sphere, e.g. the images from the transformed Berkeley dataset in Section V-A2.

Achievable segmentation accuracy (ASA) gives the highest accuracy achievable when taking superpixels as units for object segmentation. It is an upper bound measure, and is computed as the fraction of correctly labeled pixels when labeling each superpixel with the label of the ground truth segment which has the largest overlap, i.e.

$$ASA(\mathcal{G}, \mathcal{S}) = \frac{\sum_k \max_i A(s_k \cap g_i)}{\sum_i A(g_i)}. \tag{18}$$

Another important property of superpixels is structural regularity [16]–[19], which has potential advantages desired in sequential applications [20]. In this paper, we measure the structural regularity using two metrics: compactness [39] and size variation.

Compactness (Com) of a shape, sometimes called the shape factor, is a numerical quantity representing the degree to which a shape is compact [40]. A common compactness measure is the isoperimetric quotient. In two-dimensional space, the isoperimetric quotient of a shape is defined as the ratio between the area of the shape and the area of a circle having the same perimeter. In our case, we borrow the idea from isoperimetric inequality on the sphere [41] and define the isoperimetric quotient as

$$Q(s_k) = \frac{4\pi A(s_k) - A^2(s_k)}{L^2(s_k)}, \tag{19}$$

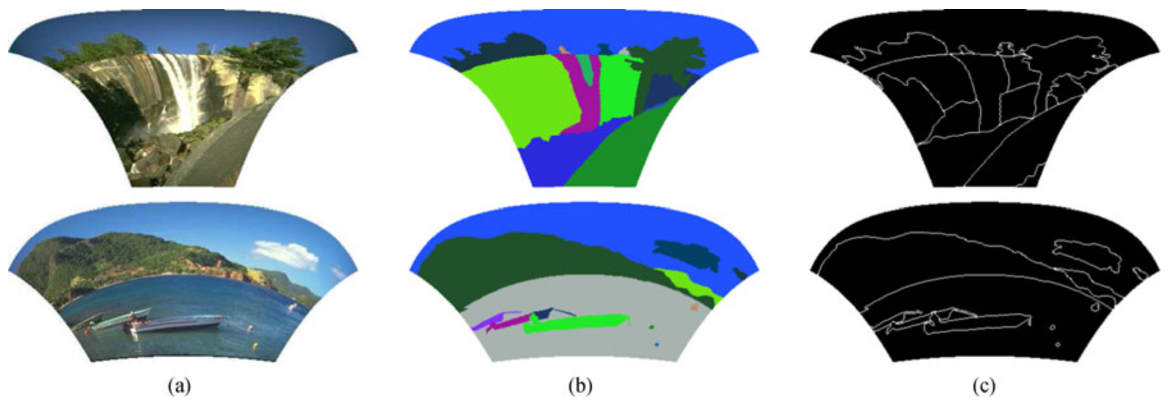


Fig. 4. Exemplary transformed images, ground truth segmentations and ground truth boundaries for images of 321 × 481 pixels (first row) and 481 × 321 pixels from Berkeley Segmentation Data Set. (Note that only the meaningful part of the spherical image and the annotation is shown here.). (a) Transformed images. (b) Ground truth segmentation. (c) Ground truth boundary.

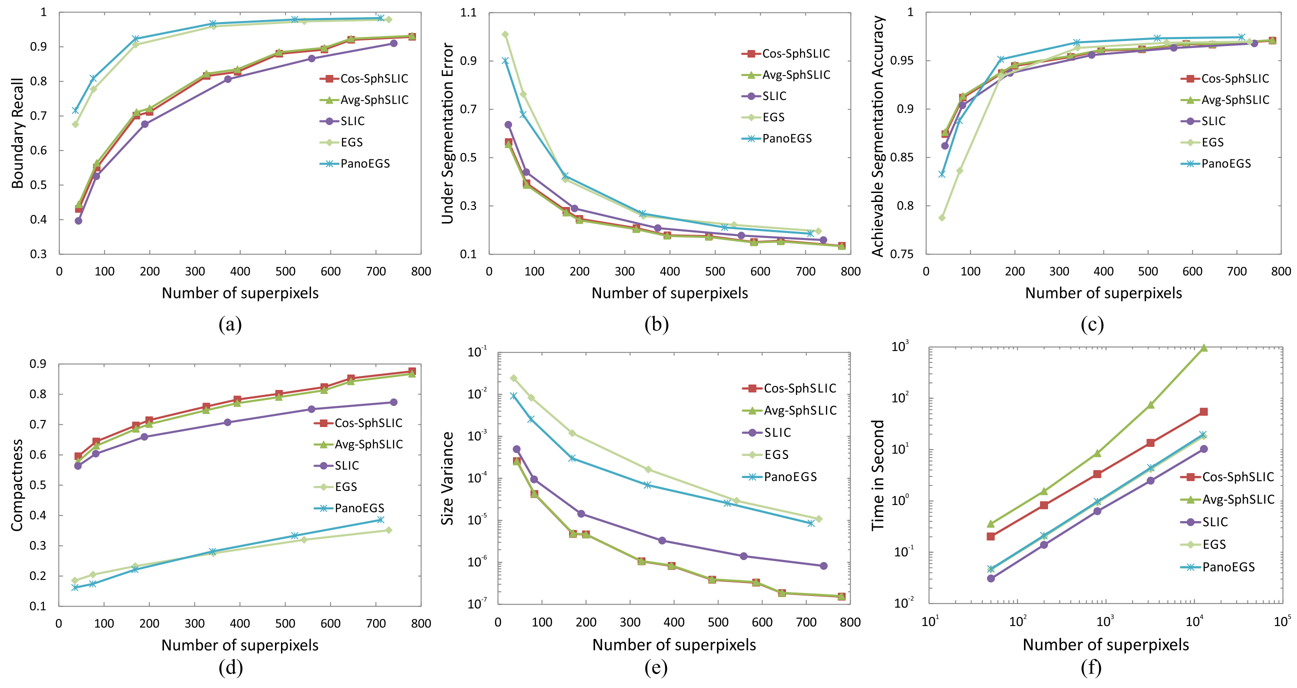


Fig. 5. Boundary recall, under segmentation error, achievable segmentation accuracy, compactness, size variation and running time of different methods on the transformed Berkeley dataset. Note that the vertical axis of the figure of size variation and the two axes of the figure of running time are in *log* scale. (a) Boundary Recall. (b) Under-Segmentation Error. (c) Achievable Segmentation Accuracy. (d) Compactness. (e) Size variation (f) Time.

where $A(s_k)$ and $L(s_k)$ are the area and perimeter of superpixel s_k respectively. In our work, $L(s_k)$ is defined as the sum of the spherical distance between adjacent boundary pixels of superpixel s_k. Given a segmentation, the compactness is the weighted mean of isoperimetric quotient of each superpixel, where weights are the areas of the superpixels, i.e.

$$Com(\mathcal{S}) = \frac{\sum_k Q(s_k)A(s_k)}{\sum_k A(s_k)}. \tag{20}$$

Size variation (SV) describes the uniformity of superpixel size. In this paper, the size variation is defined as the variance of superpixel areas $A(s_k)$. For convenience, the superpixel area is divided by the sphere size, which gives the normalized area.

2) Experiment on the Transformed Berkeley Dataset: Berkeley Segmentation Dataset [42] is the most commonly used benchmark to quantitatively evaluate different superpixel algorithms. This dataset contains 500 natural images that have been manually segmented. In this section, we evaluate different algorithms by *transforming* the Berkeley dataset. Specifically, we use each image in the dataset as a texture to render a sphere. Then the rendered sphere is flatten to a spherical image by equirectangular projection. To simulate the distortions of spherical image, we intentionally assume each image has 90° field of view and position the texture image near the polar region of the sphere. The segmentation and boundary ground truth of the original dataset are transformed in the same way. Exemplar transformed images and corresponding ground

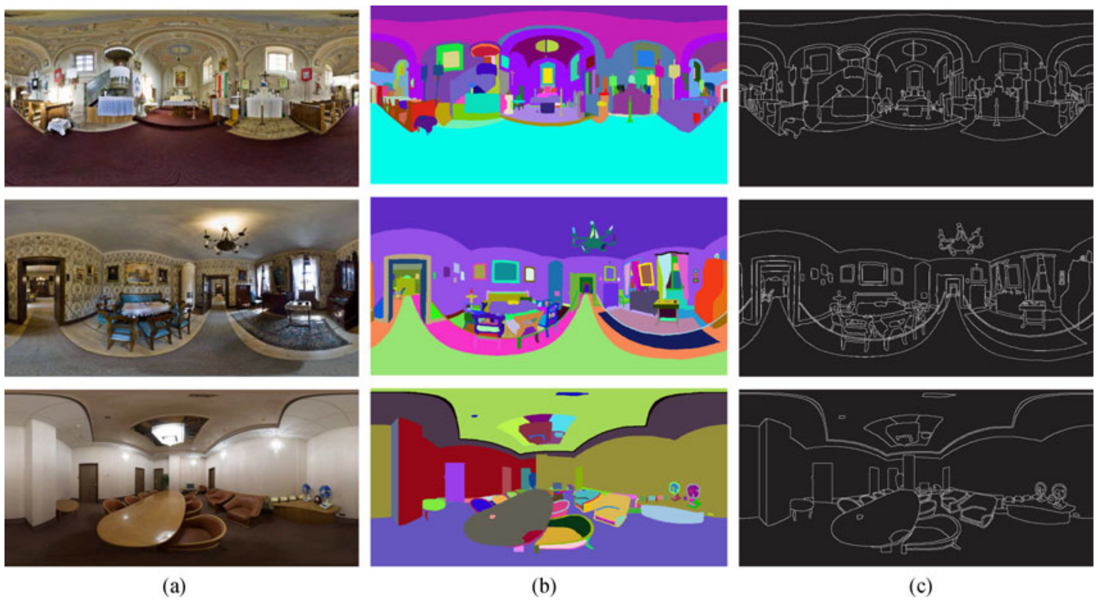


Fig. 6. Exemplary images, segmentations and image boundaries from our panorama segmentation dataset. (a) Spherical Image. (b) Segmentation. (c) Boundaries.

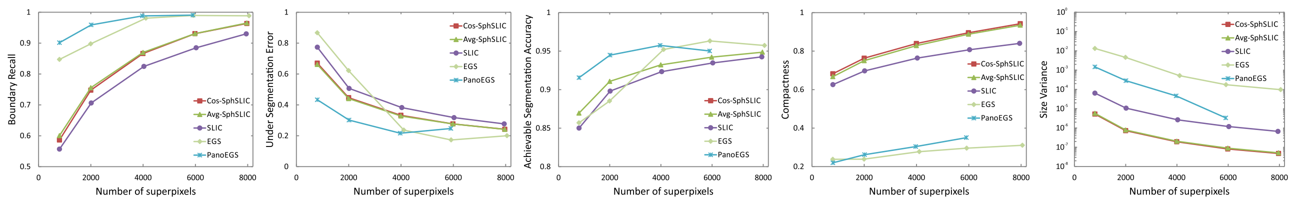


Fig. 7. The performance of different methods on real panorama segmentation dataset. Note that the vertical axis of the figure of size variation is in log scale.

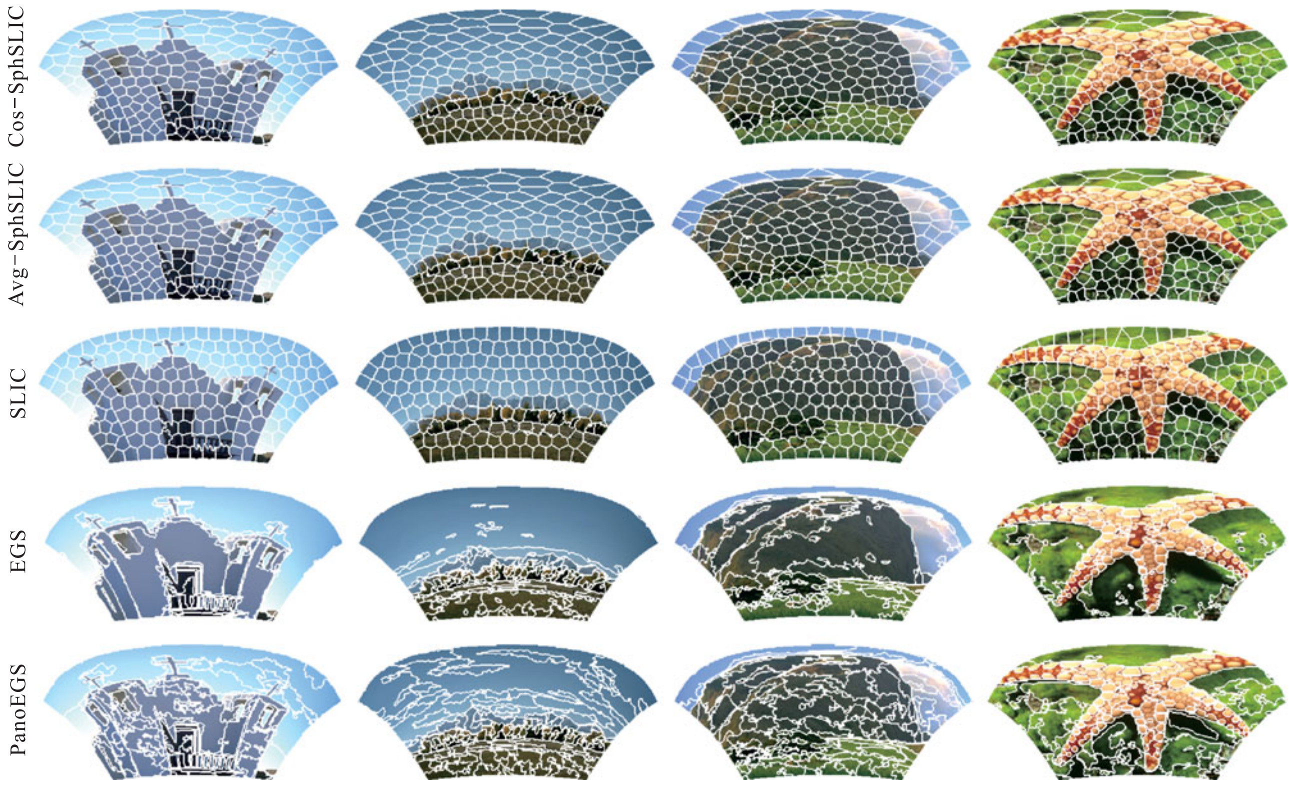


Fig. 8. Visual comparison of different methods on the transformed Berkeley segmentation data set.

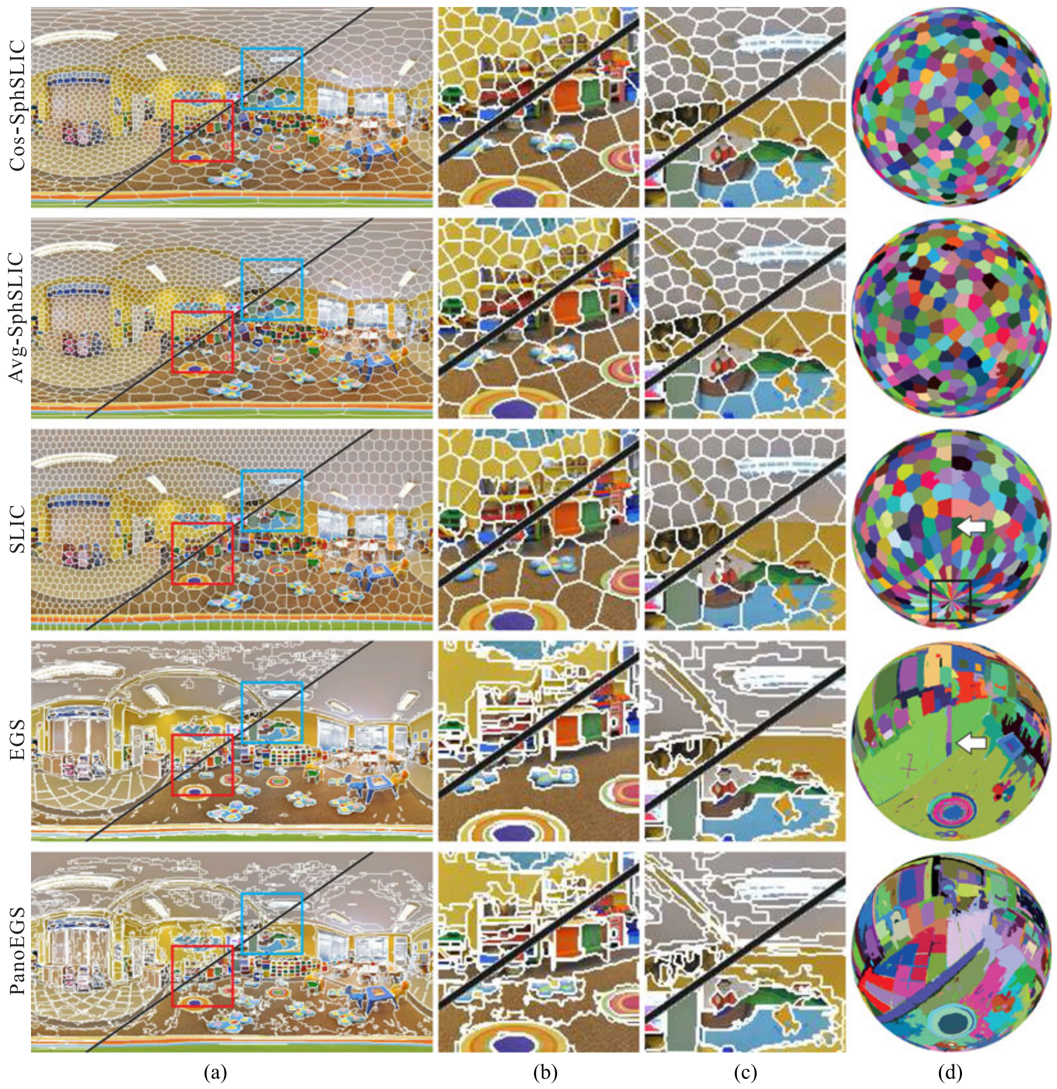


Fig. 9. Visual comparison of superpixels produced by various methods: (a) spherical images segmentation results of approximately 1953 and 678 superpixels, (b) and (c) zoom-in results, (d) results mapped to the sphere. SLIC and EGS cannot reserve the coherence across image boundaries and there are noticeable seams denoted by the arrows. The superpixels generated by SLIC in the box are stuck, because it does not consider the geometry of spherical images.

truth segmentations and boundaries are shown in Fig. 4. We can see that the original images are stretched because of the transformation.

Fig. 5 gives the performance of different algorithms on the transformed Berkeley dataset. We can see that Cos-SphSLIC and Avg-SphSLIC can obtain similar performance, which are consistently better than that of SLIC. From Fig. 5(a) we can get that the graph-based methods, i.e. EGS and PanoEGS, can achieve better boundary recall than our methods and SLIC. This is because they define a predicate for measuring the evidence for a boundary between two regions. These two methods also get similar performance, which reflects that the change of adjacency relationship in PanoEGS does not have much impact on the boundary recall of the algorithm. The under segmentation error of each method is plotted in Fig. 5(b). Among the five methods, our two methods have the minimum under segmentation errors. SLIC gets better performance than EGS for this metric, which is the same as in the planar case. When the number of superpixels is small, PanoEGS, which assigns different weights to different pixels, gives less under segmentation error than original EGS. Fig. 5(c) shows the achievable segmentation accuracy of different methods. When there are less than 100 superpixels, the performance of graph-based methods is lower than our methods and SLIC. This is because these two graph-based methods will generate relatively large superpixels that across multiple ground truth segments. When the number of superpixes exceeds 400, the five algorithms get almost the same performance. For compactness, our two methods can obtain the best performance as shown in Fig. 5(d). Although SLIC can generate regular superpixels for planar images, it does not consider the geometry of the spherical images and has a little lower performance than our methods. Because EGS and PanoEGS do not offer an explicit control over the amount of superpixels or their compactness, they offer the worst performance. Fig. 5(e) shows the superpixel size variation of different methods, which illustrates that our method can generate more uniformly sized superpixels. The two graph-based methods do not consider the spatial relationship between pixels, and generate irregular shaped superpixels.

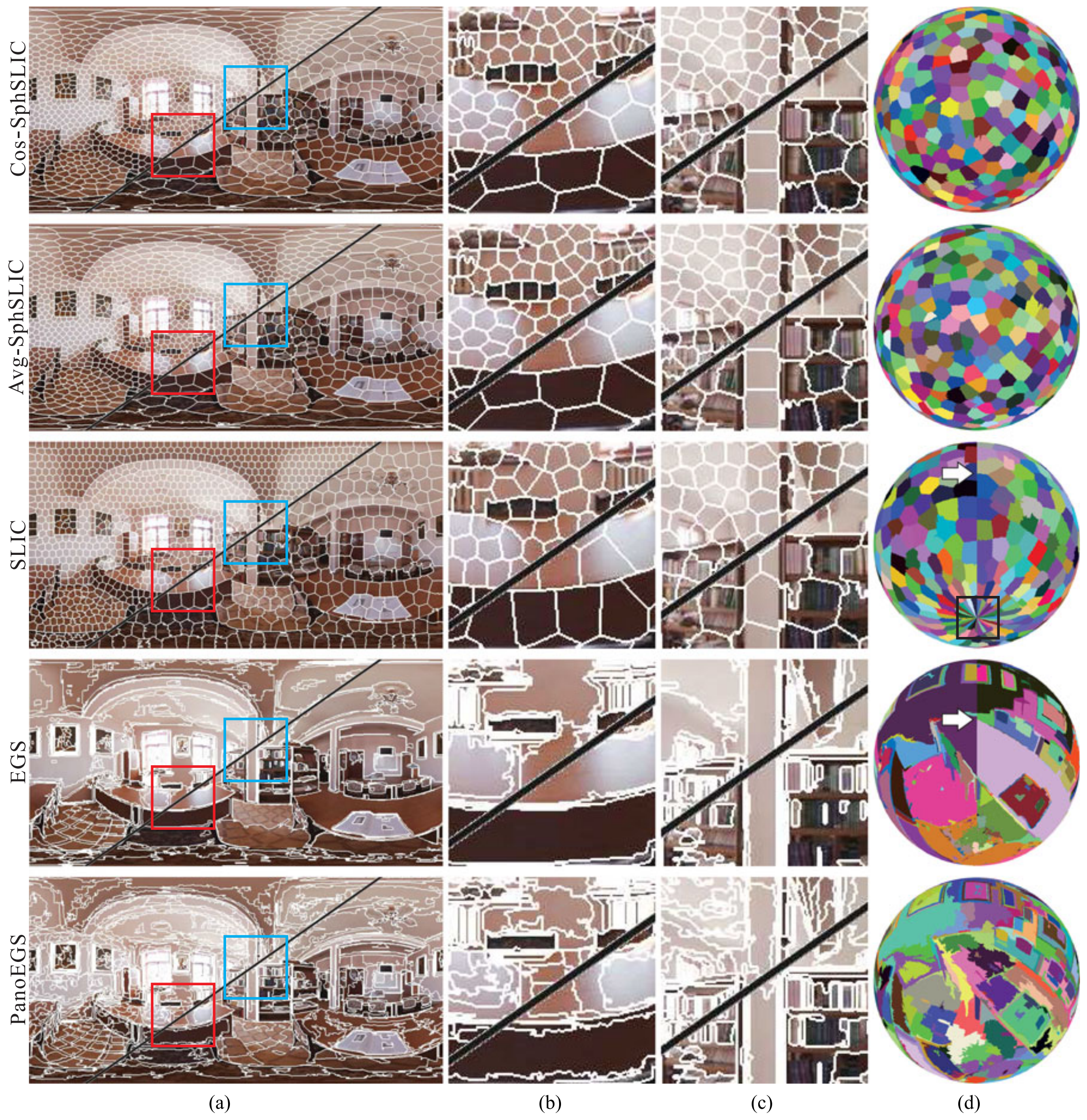


Fig. 10. Visual comparison of superpixels produced by various methods: (a) spherical images segmentation results of approximately 1875 and 524 superpixels, (b) and (c) zoom-in results, (d) results mapped to the sphere. SLIC and EGS cannot reserve the coherence across image boundaries and there are noticeable seams denoted by the arrows. The superpixels generated by SLIC in the box are stuck, because it does not consider the geometry of spherical images.

3) Experiment on Panorama Segmentation Dataset: The widely known spherical image dataset is SUN360 [43], which is initially constructed for scene recognition problem. SUN360 dataset contains $360^\circ \times 180^\circ$ panoramas without annotations, so it can not be used. Although Zhang *et al.* [44] have released a panorama dataset with annotations, these annotations are not fine enough for evaluation purpose. As each spherical image captures all the surrounding environment, constructing spherical image segmentation benchmark is really a challenging task.

In this work, we have collected a small panorama segmentation dataset. This dataset contains 12 annotated panoramas, some of which is shown in Fig. 6. We then apply different methods on this dataset and compute their performance. From Fig. 7 we can see that the result is similar to that of the experiment on the transformed Berkeley dataset, except that PanoEGS gets the best under segmentation error performance. We think this may be due to the fact that the annotations of our dataset are not as fine as those of Berkeley dataset and the capacity of our dataset is much smaller.

B. Qualitative Comparison

Fig. 8 gives the qualitative comparison of different algorithms on the transformed BSD dataset. We can see our method can achieve a little better boundary adherence than SLIC. Although graph-based methods, i.e. EGS and PanoEGS, can produce superpixels that adhere more tightly to image boundaries, these superpixels have less regular size and shape. In contrast, our

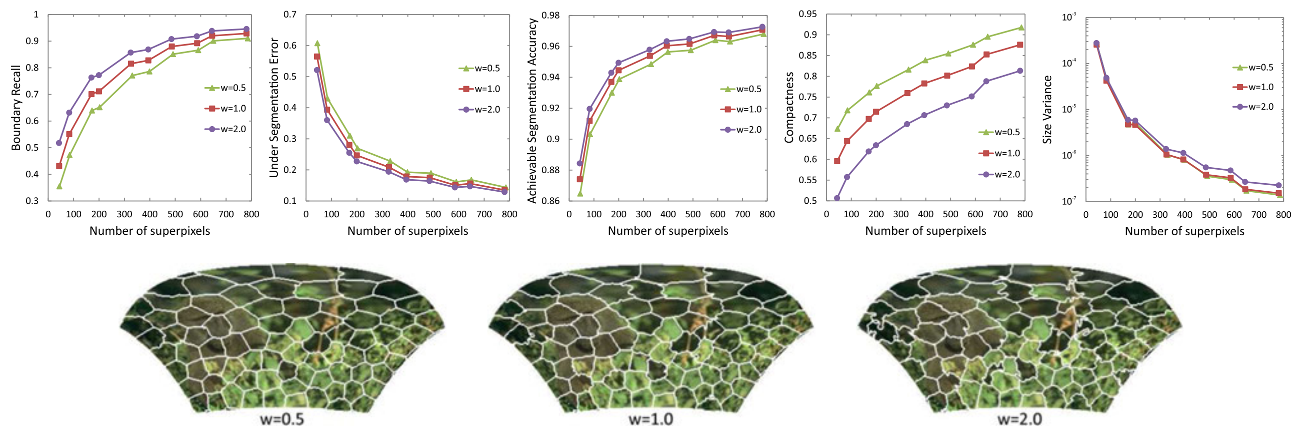


Fig. 11. The comparison of Cos-SphSLIC with different weight parameters: the quantitative performance (top) and one exemplary segmentation result (bottom).

two methods and SLIC can generate superpixels with nearly the same size.

To more thoroughly compare different methods, we select full 360° panoramas from SUN360 dataset [43]. Figs. 9 and 10 give two exemplar images, which are segmented into different number of superpixels with various methods. Due to explicit consideration of the geometry for spherical images, the superpixels generated by our method appear to be larger if they are closer to the top or bottom boundary of the spherical image. This agrees with the fact that the polar region of spherical image is over sampled and has more distortions. Because the polar region does not have many details, our method can get similar performance to SLIC with fewer superpixels as shown in the zoom-in images. Contrasted with polar region, more superpixels are extracted for central region by our method and better performance is achieved than SLIC. This explains why our method can get higher boundary recall than SLIC. To show another advantage of our method, we map the superpixel segmentation result to the image sphere as shown in Figs. 9(d) and 10(d), in which each superpixel is represented by a random color. The sphere is rotated so that we can see the south pole of the image sphere and the meridian corresponding to the left and right boundary of the spherical image. The mapped results show that the planar algorithms can not reserve the coherence across image boundaries and there are noticeable seams, which are denoted by arrows. And the superpixels near the polar region are stuck together after the segmentation results are mapped to the sphere. Another fact we can get is that our method can produce the most regular spherical superpixels. Even if the result of SLIC appears to be more regular on the equirectangular projected images, the superpixels are stretched on the sphere, such as the region bounded by the box.

C. Timing Performance

To compare the speed of different methods, we segment images of increasing size on a computer with Intel 2.50 GHz CPU and 8 GB RAM. For the images of each size, approximately the same number of superpixels are generated by each method. In our experiment, we produce about 50, 200, 800, 3200 and 12800 superpixels for spherical images with resolution 256×128, 512×256, 1024×512, 2048×1024 and 4096×2048 pixels respectively. The time values are averaged over 10 runs. In Fig. 5(f), we plot the running time required for the various methods to produce different number of superpixels. Theoretically, Cos-SphSLIC has the same complexity as SLIC. However, it takes more running time, because it involves the time consuming trigonometric functions in the assignment and update steps. The update step of Avg-SphSLIC is essentially an optimization algorithm, hence its running time is longer. EGS and PanoEGS are slower than SLIC, while faster than Cos-SphSLIC. Note that PanoEGS is a little slower than EGS. This is because it includes additional adjacency relations compared with EGS, and a post-process is applied to remove thin superpixels.

Although we have proposed two distance measures in Section IV-C, we can see that these two methods have almost the same performance from Fig. 5(a)–(e). However Avg-SphSLIC takes much more time to adjust the cluster center than Cos-SphSLIC as shown in Fig 5(f). Therefore, although the spherical distance is a more natural measure for the distance between two points on the sphere, we always use Cos-SphSLIC in practice.

D. Discussion

As shown in (11), our method has a parameter w_c, which can be used to adjust the importance between color similarity and spatial proximity. To investigate the effect of this parameter, we additionally test the quantitative performance of Cos-SpSLIC when w_c equals 0.5 and 2.0. Fig. 11 gives the comparison of Cos-SphSLIC with different parameters. From the figure we can see that a larger parameter gives better image boundary adherence, i.e. higher boundary recall, lower under-segmentation error and higher achievable segmentation accuracy. However it also sacrifices structural regularities, and gets lower compactness and higher superpixel size variance. Another fact we can get is that although boundary recall becomes higher when we use a relatively larger w_c, the under-segmentation error does not increase correspondingly. This is different from the behavior of graph-based superpixel segmentation methods. The main reason is there is spatial component in our distance measure, which can prevent from generating very large or small superpixels. Exemplary segmentation results on the transformed Berkeley dataset using different parameters are also given in Fig. 11.

VI. Conclusion

In this paper, we propose a superpixel segmentation algorithm for spherical images, which explicitly considers the geometry for the spherical images. Our approach initializes the superpixel centers using Hammersley points sampled on the sphere and takes cosine dissimilarity as the spatial part of distance measure when assigning each pixel to its nearest superpixel and updating the superpixel centers. For quantitative evaluations, we warp the images of widely used Berkeley segmentation dataset and transform them to spherical ones. We also collect a small panorama segmentation dataset. Experimental results show that our method can get better performance in terms of boundary adherence and structural regularities. Our method can also reserve the superpixel coherence across image boundaries and generate closed superpixel segments.

Because of the properties of spherical superpixels, we will investigate the application of spherical superpixels in the field of image based rendering [45] and 3D reconstruction [46] in the future. Another direction of future work involves collecting a large and more fine-grained panorama segmentation dataset, which is suitable for quantitative evaluation of spherical superpixel algorithms.

Appendix A
The Area of Pixel Set

Given a sphere of radius r, the surface area of a quadrangle bounded by the parallels ϕ_1 and ϕ_2 and the meridians θ_1 and θ_2 can be obtained by integration

$$\int_{\theta_1}^{\theta_2}\int_{\phi_1}^{\phi_2} r^2 \sin\phi d\theta d\phi = r^2(\cos\phi_1 - \cos\phi_2)(\theta_2 - \theta_1). \tag{21}$$

For the equirectangular projected spherical image, each pixel p is a quadrangle bounded by the parallels $\phi_1 = \frac{y\pi}{h}$ and $\phi_2 = \frac{(y+1)\pi}{h}$ and the meridians $\theta_1 = \frac{2x\pi}{w}$ and $\theta_2 = \frac{2(x+1)\pi}{w}$, where $[x\ y]$ is the coordinate of the pixel and $[w\ h]$ is the resolution of the image. According to (21), the surface area of pixel p then is

$$A(p) = r^2\left(\cos\frac{y\pi}{h} - \cos\frac{(y+1)\pi}{h}\right)\frac{2\pi}{w}. \tag{22}$$

Correspondingly, the area of segment g_i and superpixel s_k is $A(g_i) = \sum_{p\in g_i} A(p)$ and $A(s_k) = \sum_{p\in s_k} A(p)$ respectively.

References

[1] H. Liang, J. Yuan, and D. Thalmann, "Parsing the hand in depth images," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1241–1253, Aug. 2014.

[2] C. Wang, Z. Liu, and S. C. Chan, "Superpixel-based hand gesture recognition with kinect depth camera," *IEEE Trans. Multimedia*, vol. 17, no. 1, pp. 29–39, Jan. 2015.

[3] J. Tighe and S. Lazebnik, "Superparsing: Scalable nonparametric image parsing with superpixels," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 352–365.

[4] S. Liu *et al.*, "Fashion parsing with weak color-category labels," *IEEE Trans. Multimedia*, vol. 16, no. 1, pp. 253–265, Jan. 2014.

[5] C. L. Zitnick and S. B. Kang, "Stereo for image-based rendering using image over-segmentation," *Int. J. Comput. Vis.*, vol. 75, no. 1, pp. 49–65, 2007.

[6] G. Chaurasia, S. Duchêne, O. Sorkine-Hornung, and G. Drettakis, "Depth synthesis and local warps for plausible image-based navigation," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 1–12, 2013.

[7] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller, "Multiple hypothesis video segmentation from superpixel flows," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 268–281.

[8] L. Zhang *et al.*, "Representative discovery of structure cues for weakly-supervised image segmentation," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 470–479, Feb. 2014.

[9] H. Jiang, G. Zhang, H. Wang, and H. Bao, "Spatio-temporal video segmentation of static scenes and its applications," *IEEE Trans. Multimedia*, vol. 17, no. 1, pp. 3–15, Jan. 2015.

[10] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 670–677.

[11] L. Li, W. Feng, L. Wan, and J. Zhang, "Maximum cohesive grid of superpixels for fast object localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2013, pp. 3174–3181.

[12] J. Lei *et al.*, "A universal framework for salient object detection," *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1783–1795, Sep. 2016.

[13] J. G. Yu, G. S. Xia, C. Gao, and A. Samal, "A computational model for object-based visual saliency: Spreading attention along gestalt cues," *IEEE Trans. Multimedia*, vol. 18, no. 2, pp. 273–286, Feb. 2016.

[14] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[15] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, 2004.

[16] A. Levinshtein *et al.*, "Turbopixels: Fast superpixels using geometric flows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2290–2297, Dec. 2009.

[17] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 211–224.

[18] G. Zeng, P. Wang, J. Wang, R. Gan, and H. Zha, "Structure-sensitive superpixels via geodesic distance," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 447–454.

[19] R. Achanta *et al.*, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[20] H. Fu, X. Cao, D. Tang, Y. Han, and D. Xu, "Regularity preserved superpixels and supervoxels," *IEEE Trans. Multimedia*, vol. 16, no. 4, pp. 1165–1175, Jun. 2014.

[21] B. Micusik and J. Kosecka, "Piecewise planar city 3d modeling from street view panoramic sequences," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2009, pp. 2906–2912.

[22] J. Košecka, "Detecting changes in images of street scenes," in *Proc. Asian Conf. Comput. Vis.*, 2013, pp. 590–601.

[23] R. Cabral and Y. Furukawa, "Piecewise planar and compact floorplan reconstruction from images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2014, pp. 628–635.

[24] K. Sakurada and T. Okatani, "Change detection from a street image pair using cnn features and superpixel segmentation," in *Proc. Brit. Mach. Vis. Conf.*, 2015, pp. 61.1–61.12.

[25] D. Zorin and A. H. Barr, "Correction of geometric perceptual distortions in pictures," in *Proc. SIGGRAPH*, 1995, pp. 257–264.

[26] J. J. Cui and W. Freeden, "Equidistribution on the sphere," *SIAM J. Sci. Comput.*, vol. 18, no. 2, pp. 595–609, 1997.

[27] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 13–26.

[28] H. Yang and H. Zhang, "Efficient 3d room shape recovery from a single panorama," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2016, pp. 5422–5430.

[29] Q. Zhao, L. Wan, and J. Zhang, "Spherical superpixel segmentation," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2016, pp. 1–6.

[30] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 1–15.

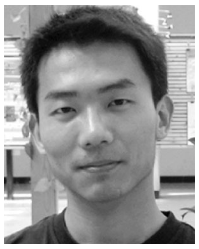
[31] T. T. Wong, W. S. Luk, and P. A. Heng, "Sampling with Hammersley and Halton points," *ACM J. Graph. Tools*, vol. 2, no. 2, pp. 9–24, 1997.

[32] S. Tezuka, *Uniform Random Numbers: Theory and Practice*. Berlin, Germany: Springer-Verlag, 1995.

[33] Q. Zhao, W. Feng, L. Wan, and J. Zhang, "SPHORB: A fast and robust binary feature on the sphere," *Int. J. Comput. Vis.*, vol. 113, no. 2, pp. 143–159, 2015.

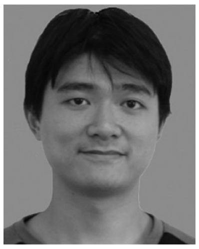
[34] K. Hornik, I. Feinerer, M. Kober, and C. Buchta, "Spherical k-means clustering," *J. Statist. Softw.*, vol. 50, no. 1, pp. 1–22, 2012.

[35] S. R. Buss and J. P. Fillmore, "Spherical averages and applications to spherical splines and interpolation," *ACM Trans. Graph.*, vol. 20, no. 2, pp. 95–126, 2001.
[36] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2003, pp. 10–17.
[37] M. Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2011, pp. 2097–2104.
[38] S. Nowozin, P. V. Gehler, and C. H. Lampert, "On parameter learning in CRF-based approaches to object class image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 98–111.
[39] A. Schick, M. Fischer, and R. Stiefelhagen, "An evaluation of the compactness of superpixels," *Pattern Recogn. Lett.*, vol. 43, pp. 71–80, 2014.
[40] WIKIPEDIA, "Compactness measure of a shape." [Online]. Available: https://en.wikipedia.org/wiki/Compactness_measure_of_a_shape
[41] R. Osserman, "The isoperimetric inequality," *Bull. Amer. Math. Soc.*, vol. 84, no. 6, pp. 1182–1238, 1978.
[42] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2001, pp. 416–423.
[43] J. X. Xiao, K. A. Ehinger, A. Oliva, and A. Torralba, "Recognizing scene viewpoint using panoramic place representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2012, pp. 2695–2702.
[44] Y. D. Zhang, S. R. Song, P. Tan, and J. X. Xiao, "Panocontext: A whole-room 3d context model for panoramic scene understanding," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 668–686.
[45] Q. Zhao, L. Wan, W. Feng, J. Zhang, and T. T. Wong, "Cube2video: Navigate between cubic panoramas in real-time," *IEEE Trans. Multimedia*, vol. 15, no. 8, pp. 1745–1754, Dec. 2013.
[46] A. Pagani and D. Stricker, "Structure from motion using full spherical panoramic cameras," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2011, pp. 375–382.

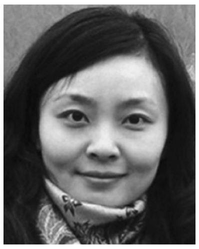


Qiang Zhao received the B.Eng. degree in software engineering and Ph.D. degree in computer science and technology from Tianjin University, Tianjin, China, in 2009 and 2016, respectively. He is currently an Assistant Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His main research interests include image-based rendering, feature extraction, and light field image processing.

Feng Dai (M'13) received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2008. He is currently an Associate Professor in the Multimedia Computing Group, Advanced Research Laboratory, Institute of Computing Technology, Chinese Academy of Sciences. His research interests include image/video processing, computational imaging, and computer vision.



Yike Ma (M'13) received the B.S. degree from Harbin Institute of Technology, Harbin, China, in 2003, and Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2011. He is currently an Associate Professor with the Institute of Computing Technology, Chinese Academy of Sciences. His interests include computational photography, image processing, and algorithm parallel optimization.



Liang Wan (M'08) received the B.Eng. and M.Eng. degrees in computer science and engineering from Northwestern Polytechnical University, Xi'an, China, in 2000 and 2003, respectively, and the Ph.D. degree in computer science and engineering from Chinese University of Hong Kong, Hong Kong, in 2007. She is currently an Associate Professor in the School of Computer Software, Tianjin University, Tianjin, China. Her research interests include intelligent image synthesis, including image-based rendering, image navigation, precomputed lighting, and panoramic image processing.

Jiawan Zhang (M'06) received the B.Sci., M.Phil., and Ph.D. degrees in computer science from Tianjin University, Tianjin, China, in 1997, 2001, and 2004, respectively. He is currently a Professor in the School of Computer Software, Tianjin University. His main research interests include computer graphics, visual analytics, and digital culture heritage. Prof. Zhang is a Senior Member of China Society of Image and Graphics, a Senior Member of China Computer Federation, and a co-chair of Tianjin Society of Image and Graphics.

Yongdong Zhang (M'08–SM'13) received the Ph.D. degree in electronics engineering from Tianjin University, Tianjin, China, in 2002. He is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. He has authored more than 100 refereed journal and conference papers. His current research interests include multimedia content analysis and understanding, multimedia content security, video encoding, and streaming media technology. Prof. Zhang was a recipient of the best paper awards in PCM 2013, ICIMCS 2013, and ICME 2010, and the Best Paper Candidate in ICME 2011. He serves as an Editorial Board Member of the *Multimedia Systems Journal* and the IEEE TRANSACTIONS ON MULTIMEDIA.